Embedded Syslog Manager コンフィギュレーション ガイド

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255 （フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/

【注意】シスコ製品をご使用になる前に、安全上の注意（**www.cisco.com/jp/go/safety_warning/**）をご確認ください。本書は、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。また、契約等の記述については、弊社販売パートナー、または、弊社担当者にご確認ください。

このマニュアルに記載されている仕様および製品に関する情報は、予告なしに変更されることがあります。このマニュアルに記載されている表現、情報、および推奨事項は、すべて正確であると考えていますが、明示的であれ黙示的であれ、一切の保証の責任を負わないものとします。このマニュアルに記載されている製品の使用は、すべてユーザ側の責任になります。

対象製品のソフトウェア ライセンスおよび限定保証は、製品に添付された『Information Packet』に記載されています。添付されていない場合には、代理店にご連絡ください。

シスコが導入する TCP ヘッダー圧縮は、カリフォルニア大学バークレー校（UCB）により、UNIX オペレーティング システムの UCB パブリック ドメイン バージョンの一部として開発されたプログラムを適応したものです。All rights reserved.Copyright © 1981, Regents of the University of California.

ここに記載されている他のいかなる保証にもよらず、各社のすべてのマニュアルおよびソフトウェアは、障害も含めて「現状のまま」として提供されます。シスコおよびこれら各社は、商品性の保証、特定目的への準拠の保証、および権利を侵害しないことに関する保証、あるいは取引過程、使用、取引慣行によって発生する保証をはじめとする、明示されたまたは黙示された一切の保証の責任を負わないものとします。

いかなる場合においても、シスコおよびその供給者は、このマニュアルの使用または使用できないことによって発生する利益の損失やデータの損傷をはじめとする、間接的、派生的、偶発的、あるいは特殊な損害について、あらゆる可能性がシスコまたはその供給者に知らされていても、それらに対する責任を一切負わないものとします。

このマニュアルで使用している IP アドレスおよび電話番号は、実際のアドレスおよび電話番号を示すものではありません。マニュアル内の例、コマンド出力、ネットワーク トポロジ図、およびその他の図は、説明のみを目的として使用されています。説明の中に実際のアドレスおよび電話番号が使用されていたとしても、それは意図的なものではなく、偶然の一致によるものです。

CiscoおよびCisco ロゴは、シスコまたはその関連会社の米国およびその他の国における商標または登録商標です。シスコの商標の一覧は、http://www.cisco.com/go/trademarksでご確認いただけます。掲載されている第三者の商標はそれぞれの権利者の財産です。「パートナー」または「partner」という用語の使用はシスコと他社との間のパートナーシップ関係を意味するものではありません。(1110R)

© 2017 Cisco Systems, Inc. All rights reserved.

目 次

最初にお読みください 1

Embedded Syslog Manager（ESM） 3

機能情報の確認 3

Embedded Syslog Manager の制約事項 4

Embedded Syslog Manager について 4

システム メッセージ ロギング 4

システム ロギング メッセージの形式 4

Embedded Syslog Manager の利点 5

syslog フィルタ モジュール 5

Embedded Syslog Manager の使用方法 6

ESM syslog フィルタ モジュールの書き込み 6

ESM フィルタ プロセス 6

syslog フィルタ モジュールの入力 7

標準的な ESM フィルタ処理 7

バックグラウンド ESM フィルタの処理 8

次の作業 10

Embedded Syslog Manager の設定 10

Embedded Syslog Manager の設定例 14

例：Embedded Syslog Manager の設定例 14

例：syslog フィルタ モジュール 15

例：重大度のエスカレーション 15

例：メッセージのカウント 15

例：XML タギング 18

例：SMTP ベースの電子メール アラート 20

例：ストリーム 21

例：送信元 IP タギング 21

Embedded Syslog Manager に関する追加情報 22

Embedded Syslog Manager の機能情報 24
用語集 25
ローカル不揮発性ストレージへのロギング 27
機能情報の確認 27
ローカル不揮発性ストレージへのロギングの前提条件 28
ローカル不揮発性ストレージへのロギングの制約事項 28
ローカル不揮発性ストレージへのロギングに関する情報 28
システム ロギング メッセージ 28
ローカル不揮発性ストレージへのロギングの設定方法 29
ブートフラッシュまたはハードディスクへのロギング メッセージの書き込み 29
外部ディスクへのロギング メッセージのコピー 30
ローカル不揮発性ストレージへのロギングの設定例 31
例：ブートフラッシュまたはハードディスクへのロギング メッセージの書き込み 31
例：外部ディスクへのロギング メッセージのコピー 31
その他の参考資料 31
ローカル不揮発性ストレージへのロギングの機能情報 32
syslog の信頼性の高い伝送およびフィルタリング 35
機能情報の確認 35
syslog の信頼性の高い伝送およびフィルタリングの前提条件 36
syslog の信頼性の高い伝送およびフィルタリングの制約事項 36
syslog の信頼性の高い伝送およびフィルタリングに関する情報 37
BEEP 転送のサポート 37
syslog メッセージ 38
syslog セッション 38
複数の syslog セッション 40
メッセージ ディスクリミネータ 41
レート制限 42
syslog の信頼性の高い伝送およびフィルタリングの利点 42
syslog の信頼性の高い伝送およびフィルタリングの設定方法 43
メッセージ ディスクリミネータの作成 43
メッセージ ディスクリミネータのロギング バッファとの関連付け 44

メッセージ ディスクリミネータのコンソール端末との関連付け 45
メッセージ ディスクリミネータの端末回線との関連付け 46
メッセージ カウンタのイネーブル化 47
BEEP セッションの追加と削除 48
syslog の信頼性の高い伝送およびフィルタリングの設定例 50
転送とロギングの設定例 50
syslog トランザクションの VRF 対応送信元インターフェイスに関する追加情報 50
syslog の信頼性の高い伝送およびフィルタリングの機能情報 51

第 1 章

最初にお読みください

Cisco IOS XE 16 に関する重要な情報

現行の Cisco IOS XE リリース 3.7.0E（Catalyst スイッチ用）および Cisco IOS XE リリース 3.17S（アクセスおよびエッジルーティング用）の 2 つのリリースは、1 つのバージョンの統合されたリリース（Cisco IOS XE 16）へと発展しています。これにより、スイッチングおよびルーティング ポートフォリオの幅広い範囲のアクセスおよびエッジ製品に 1 つのリリースで対応できます。

（注） 技術設定ガイドの機能情報の表には、機能が導入された時期が示されています。その他のプラットフォームでその機能がサポートされた時期については示されていない場合があります。特定の機能がご使用のプラットフォームでサポートされているかどうかを特定するには、製品のランディング ページに示されている技術設定ガイドを参照してください。技術設定ガイドが製品のランディング ページに表示されている場合は、その機能がプラットフォームでサポートされていることを示します。

第 2 章

Embedded Syslog Manager（ESM）

Embedded Syslog Manager（ESM）機能は、システム メッセージ ロガーによって伝送される前にシステム ロギング メッセージをフィルタリング、拡大、相互関連付け、ルーティング、カスタマイズできるようにするプログラマブル フレームワークを提供します。

- 機能情報の確認，3 ページ
- Embedded Syslog Manager の制約事項，4 ページ
- Embedded Syslog Manager について，4 ページ
- Embedded Syslog Manager の使用方法，6 ページ
- Embedded Syslog Manager の設定例，14 ページ
- Embedded Syslog Manager に関する追加情報，22 ページ
- Embedded Syslog Manager の機能情報，24 ページ
- 用語集，25 ページ

機能情報の確認

ご使用のソフトウェア リリースでは、このモジュールで説明されるすべての機能がサポートされているとは限りません。最新の機能情報および警告については、Bug Search Tool およびご使用のプラットフォームおよびソフトウェア リリースのリリース ノートを参照してください。このモジュールに記載されている機能の詳細を検索し、各機能がサポートされているリリースのリストを確認する場合は、このモジュールの最後にある機能情報の表を参照してください。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

Embedded Syslog Manager の制約事項

Embedded Syslog Manager（ESM）は Tcl 8.3.4 サブシステムに依存します。これは、ESM フィルタが Tool Command Language（Tcl）で作成されるためです。ESM は、Tcl バージョン 8.3.4 以降をサポートするイメージでのみ使用可能です。Tcl 8.3.4 サポートが追加されるかどうかはリリースによって異なります。

ESM フィルタは Tcl で書かれています。

ESM フィルタリングは、SNMP「履歴」ロギングには適用できません。したがって、**logginghistory** および **snmp-serverenabletrapssyslog** コマンドを使ってロギングされるメッセージには ESM フィルタリングが適用されません。

Embedded Syslog Manager について

システム メッセージ ロギング

Embedded Syslog Manager を導入すると、システム メッセージを標準メッセージ、XML 形式のメッセージ、または ESM でフィルタリングされたメッセージとして、別個にロギングできるようになります。これらの出力は、従来のあらゆる syslog ターゲットに送信できます。たとえば、コンソール接続への標準ロギング、XML 形式のメッセージのバッファへのロギング、および ESM でフィルタリングされたメッセージのモニタへのロギングをイネーブルにできます。同様に、各タイプの出力は異なるリモート ホストに送信できます。別個のロギング プロセスの利点は、たとえば ESM フィルタ モジュールに問題がある場合に標準のロギングが影響を受けないことです。

システム ロギング メッセージの形式

システム ロギング メッセージは、次の形式で表示されます。

%<ファシリティ>-<重大度>-<ニーモニック>: <メッセージ テキスト>

以下に、システム ロギング メッセージの例を示します。

```
%LINK-5-CHANGED: Interface Serial3/3, changed state to administratively down
```

通常は、これらのメッセージの前にエラー シーケンス番号やタイムスタンプなどの追加のテキストが存在します。

<シーケンス番号>: <タイム スタンプ>:%<ファシリティ>-<重大度>-<ニーモニック>: <メッセージ テキスト>

エラー シーケンス番号とタイムスタンプの後に続くシステム ロギング メッセージの例を以下に示します。

```
000013: Mar 18 14:52:10.039:%LINK-5-CHANGED: Interface Serial3/3, changed state
```

```
to administratively down
```

（注） システム ロギング メッセージで使用されるタイム スタンプの形式は、**servicetimestamps** グローバル コンフィギュレーション モード コマンドによって決まります。**servicesequence-numbers** グローバル コンフィギュレーション コマンドは、先頭のシーケンス番号をイネーブルまたはディセーブルにします。時刻の前のアスタリスク（*）は、システム クロックが信頼できる時刻源と同期していないため、時刻が不正確である可能性があることを示しています。

Embedded Syslog Manager の利点

Cisco ソフトウェアの組み込み機能である Embedded Syslog Manager（ESM）を使用すると、送信元でのシステム メッセージ ロギングの完全な制御が可能になります。ESM に備わっているプログラマティック インターフェイスを使用すると、システム ロギングに関連する特定のニーズを満たすカスタム フィルタを作成できます。この機能の利点は次のとおりです。

- カスタマイズ：システム ロギング メッセージを扱う完全にカスタマイズ可能な処理。相互に連携する複数の syslog コレクタをサポートします。
- 主要なメッセージの重大度のエスカレーション：システム定義の重大度レベルを使用する代わりに、syslog メッセージに関する独自の重大度レベルを設定できます。
- 特定のメッセージに対象を絞る：ファシリティのタイプや重大度のタイプに基づいて、特定のメッセージまたはメッセージ タイプをさまざまな syslog コレクタにルーティングできます。
- SMTP ベースの E メール アラート：TCP ベースの syslog コレクタまたはシンプル メール転送プロトコル（SMTP）サーバなどの外部サーバに TCP を使用して通知する機能。
- メッセージの制限：デバイスレベルのイベントを関連付けることにより、syslog の「メッセージ ストーム」を制限し、管理することができます。

ESM は UDP ベースの syslog メカニズムに代わるものではなく、現在のシステム ロギング プロセスと並行して動作可能なオプションのサブシステムです。たとえば、元の syslog メッセージ ストリームをサーバ A によって収集し続けるのと同時に、フィルタリング、相互関連、その他の方法でカスタマイズされた ESM ロギング ストリームをサーバ B に送信することができます。元の syslog ストリームまたは ESM ストリームのいずれかを受信するよう、syslog メッセージの現在のすべてのターゲット（コンソール、モニタ、バッファ、および syslog ホスト リスト）を設定することができます。ESM ストリームは、それに応じて、ユーザ定義のストリームにさらに分割し、コレクタにルーティングできます。

syslog フィルタ モジュール

Embedded Syslog Manager（ESM）は syslog フィルタ モジュールを使用してシステム ロギング メッセージを処理します。syslog フィルタ モジュールは、ローカル システム メモリまたはリモート

ファイル サーバに保存される Tool Command Language（Tcl）で書かれたスクリプトです。ESM は、独自のスクリプトを書いて参照できるため、カスタマイズ可能です。

syslog フィルタ モジュールは、プレーンテキスト ファイルまたはコンパイル済みファイルとして書いて、保存できます。Tcl スクリプトの事前コンパイルは、TclPro などのツールを使用して実行できます。コンパイル済みスクリプトは編集できないため、セキュリティおよび管理された一貫性の尺度になります。

（注） Tcl スクリプト モジュールには実行可能コマンドが含まれているため、コンフィギュレーション ファイルを管理するのと同じ方法で、これらのファイルのセキュリティを管理する必要があります。

Embedded Syslog Manager の使用方法

ESM syslog フィルタ モジュールの書き込み

Embedded Syslog Manager（ESM）設定で syslog フィルタ モジュールを参照する前に、システム ロギング メッセージへの適用対象となるモジュールを書き込むか、取得する必要があります。syslog フィルタ モジュールは、ローカル システム メモリまたはリモート ファイル サーバに保存できます。syslog フィルタ モジュールを書き込む前に、次の概念を理解しておく必要があります。

ESM フィルタ プロセス

ESM がイネーブルの場合、すべてのシステム ロギング メッセージは、参照された syslog フィルタ モジュールを通して処理されます。syslog フィルタ モジュールは、フィルタ チェーンで順番に処理されます。フィルタ チェーン内の syslog フィルタ モジュールの位置は、**loggingfilter** グローバル コンフィギュレーション モード コマンドで適用された位置タグによって決定されます。位置が指定されていない場合、モジュールは、設定に追加された順番で処理されます。

各フィルタ モジュールの出力は、チェーン内の次のフィルタ モジュールの入力として使用されます。したがって、元の syslog メッセージ（::orig_msg）を含む Tcl グローバル変数は、チェーン内の次のフィルタを呼び出す前に、各フィルタの戻り値に設定されます。したがって、フィルタが NULL を戻した場合、メッセージは ESM ストリームに送信されません。すべてのフィルタがメッセージを処理した後、メッセージは、ロガーによる配信のためのキューに入れられます。

コンソール、バッファ、モニタ、および syslog ホストは、特定のメッセージ ストリーム（通常、XML、または ESM）を受信するように設定できます。syslog ホストは、ユーザ定義の番号が付いたストリームを受信するように、さらに限定できます。各ターゲットは各メッセージを検査し、そのストリーム タグに基づいてメッセージを受け入れるか、または拒否します。ESM フィルタは、Tcl グローバル変数「::stream」を変更することによってメッセージのストリーム タグを変更し、宛先ストリームを変更できます。

syslog フィルタ モジュールの入力

Embedded Syslog Manager（ESM）が有効になっている場合、システム ロギング メッセージがロギング プロセスに送信されます。システム ロギング メッセージの中、およびフォーマットされた syslog メッセージ全体に含まれる各データ エレメントが、Tcl グローバル変数として記録されます。syslog メッセージのデータ エレメントの形式は、次のとおりです。

<シーケンス番号>: <タイム スタンプ>:%<ファシリティ>-<重大度>-<ニーモニック>: <メッセージ テキスト>

メッセージ テキストには、しばしばメッセージ引数が含まれます。

また、メッセージが syslog ホストで受信されるときに「syslog カウント」番号が追加されます。

<syslog カウント>: <シーケンス番号>: <タイム スタンプ>:%<ファシリティ>-<重大度>-<ニーモニック>: <メッセージ テキスト>

次の例は、シーケンスの最初に含まれる syslog カウント番号を示しています。

以下の表に、syslog フィルタ モジュールで使用される Tcl スクリプト入力変数を示します。フィルタで扱う必要のある syslog メッセージ データは、Tcl グローバル名前空間変数として渡されます。したがって、スクリプト モジュール内で変数の前に二重のコロンを付ける必要があります。

標準的な ESM フィルタ処理

システム ロギング メッセージが生成されるたびに、syslog フィルタ モジュールがシリーズで呼び出されます。このシリーズは ::module_position 変数によって決定されますが、これは一般的にシステム設定内でモジュールが参照される順番（モジュールが設定される順番）です。

あるフィルタ モジュールの出力が、次のフィルタ モジュールへの入力になります。フィルタへの入力は Tcl グローバル名前空間変数であるため、それぞれのフィルタは、フィルタの目的に応じてこれらの変数のいずれかまたはすべてを変更できます。

後続の複数のフィルタ実行の間で Embedded Syslog Manager（ESM）フレームワークによって自動的に更新される Tcl グローバル変数は、::orig_msg 変数と ::cli_args 変数だけです。フレームワークは、自動的に ::orig_msg の値をフィルタ モジュールの戻り値に設定します。したがって、元のメッセージを変更するかフィルタリングするように設定されているフィルタでは、::orig_msg 変数の値を手動で設定しないでください。フィルタは目的の値を返すことだけが必要です。たとえば、次の 1 行の ESM フィルタ

```
return "This is my new syslog message."
```

は、受け取ったメッセージをすべて無視して出力を「This is my new syslog message」という定数文字列に常に変更します。このモジュールがチェーン内の最後のフィルタである場合、すべての ESM ターゲットはこの文字列を最終的な syslog メッセージとして受け取ります。

1 行の ESM フィルタは、

```
return ""
```

ESM ストリームへのすべての syslog メッセージをブロックします。たとえば、

```
return $::orig_msg
```

という行は、メッセージをチェーンの次のフィルタに渡すだけです。したがって、不要なメッセージを抑制するように設定されている ESM フィルタは次のように見えます。

```
if { [my_procedure_to_check_this_message] == 1 } {
     return $::orig_msg
} else {
     return ""
}
```

その設計によって、一部のフィルタは ::orig_msg 変数をまったく使用せず、そのデータ エレメントから syslog メッセージを再構築する（::format_string、::msg_args、::timestamp などを使用）場合があります。たとえば、XML タグ付きフィルタは個々のデータ エレメントにタグを付け、元のフォーマットされたメッセージを無視します。このようなモジュールが Tcl スクリプトの最初で ::orig_msg 変数を検査することは重要です。以前のフィルタで「メッセージを送信してはならない」と指示されている場合（つまり ::orig_msg が NULL の場合）、メッセージは処理されず、NULL を返します。

また、**exec** および **config** Tcl コマンドを使用して syslog フィルタ モジュールにコマンドを追加することもできます。たとえば、syslog メッセージに送信元 IP アドレスを追加する必要があり、しかも（**logging source-interface**コマンドを使って）イーサネット 2/0 インターフェイスから送信されるよう syslog メッセージが設定される場合には、スクリプト内で **exec** Tcl コマンドを次のように使用することで、モジュールの初期化中に **show interface Ethernet 2/0** コマンドを発行できます。

```
set source_ip_string [exec show ip int E2/0 | inc Internet]
puts $source_ip_string
"  Internet address is 10.4.2.63/24"
```

バックグラウンド ESM フィルタの処理

Tcl では、後の時点で処理するためにコマンドをキューに入れることができます。それを行うには **after** Tcl コマンドを使用します。このコマンドの最も一般的な使用方法は、一定の時間間隔（「相関期間」と呼ばれる）にわたって、複数のイベントを相互に関連付ける（収集して要約する）ことです。該当する期間が満了すると、フィルタは「ウェイクアップ」し、その期間に発生したイベントを計算または要約する必要があります。また、多くの場合、イベント報告用の新しい syslog メッセージを送信する必要があります。このバックグラウンド プロセスは、特定の時間の経過後、Tcl インタープリタがキューに入れられたコマンドを実行できるようにする ESM イベント ループ プロセスによって処理されます。

syslog フィルタ モジュールが相関期間を利用する必要がある場合、相関期間が満了した時点で **after** Tcl コマンドを使用して要約手順を呼び出す必要があります（「Embedded Syslog Manager の設定例」セクションにある例を参照）。バックグラウンド プロセスの実行中は通常のフィルタ チェーンの処理が行われないため、出力を生成するために、これらのフィルタで 2 つの ESM Tcl 拡張機能（**errmsg** または **esm_errmsg**）のいずれかを使用する必要があります。

バックグラウンド処理中に、**after** コマンドによってキューに入れられたコマンドは、（通常の処理のように）フィルタ チェーンのコンテキストでは実行されず、Tcl インタープリタによって連続して実行される自律的な手順となります。したがって、これらのバックグラウンド手順では通常の Tcl グローバル名前空間変数を扱うべきではありません（ただし **esm_errmsg** の使用時に次のフィルタ用にグローバル名前空間変数を設定する場合を除く）。代わりに、独自の名前空間に保存される変数を扱う必要があります。これらの変数が手順の定義の外側で宣言される場合、すべてのコールで永続的です。

errmsg Tcl コマンドの目的は、新しいメッセージを作成して配信用に送り出し、その際に他の syslog フィルタ モジュールをすべて回避することです。**errmsg** コマンドの構文は次のとおりです。

```
errmsg <severity> <stream> <message_string>
```

esm_errmsg Tcl コマンドの目的は、新しいメッセージを作成し、フィルタ チェーン内のそれ以降にある syslog フィルタ モジュールでメッセージを処理して、配信用に送り出すことです。**esm_errmsg** コマンドの構文は次のとおりです。

```
esm_errmsg <module_position>
```

errmsg() Tcl 関数と esm_errmsg() Tcl 関数の主な相違点として、**errmsg** はフィルタを無視してメッセージを配信用にキューに直接入れます。一方、**esm_errmsg** はフィルタ チェーンの後続部分に syslog メッセージを送ります。

次の例では、新しい syslog メッセージが作成され、アラート重大度 1 のタグを付けられて、設定された ESM ロギング ターゲット（ストリーム 2）に送信されます。このフィルタの目的は、30 分の相関期間中に個々の SYS-5-CONFIG メッセージを抑制し、ウィンドウ終了時に要約メッセージを送信することです。

```
errmsg 1 2 "*Jan 24 09:34:02.539: %SYS-1-CONFIG_I: There have been 12
configuration changes to the router between Jan 24 09:04:02.539 and Jan 24
09:34:01.324"
```

（それ以降にある残りのフィルタを呼び出す）**esm_errmsg** を使用するために、このバックグラウンド プロセスは **esm_errmsg** を呼び出す前に、必要な Tool Command Language（Tcl）グローバル名前空間変数を設定する必要があります。::module_position を渡すと、開始するフィルタが ESM フレームワークに通知されます。したがって、**esm_errmsg** コマンドを使用するフィルタは、バックグラウンド処理用に、（通常の処理中にグローバル名前空間変数で渡される）::module_position を独自の名前空間変数に保存する必要があります。次に例を示します。

```
proc ::my_filter_namespace::my_summary_procedure{}
{
 set ::orig_msg "*Jan 24 09:34:02.539: %SYS-1-CONFIG_I: There have been 12
configuration changes to the router between Jan 24 09:04:02.539 and Jan 24
09:34:01.324"
 set ::timestamp "*Jan 24 09:34:02.539"
 set ::severity 1
 set ::stream 2
 set ::traceback ""
 set ::pid ""
set ::process ""
set ::format_string "There have been %d configuration changes to the router
between %s and %s"
set ::msg_args {12 "Jan 24 09:04:01.539" "Jan 24 09:34:01.324"}
esm_errmsg $::my_filter_namespace::my_module_position
}
```

esm_errmsg コマンド用にすべてのグローバル名前空間変数を設定することの利点は、フィルタがモジュール式であり、ESM フレームワークでのフィルタの使用順序が重要でないことです。たとえば、ESM を宛先とするすべてのメッセージの後ろにメッセージ発信者のホスト名を付ける必要がある場合、次のように 1 行の「hostname」フィルタを作成し、フィルタ チェーンの最後に配置できます。

```
return "$::orig_msg -- $::hostname"
```

この例では、バックグラウンド処理中にいずれかのフィルタが新しいメッセージを生成し、**errmsg** ではなく **esm_errmsg** を使用する場合、これらのメッセージの後ろに明確にホスト名が付けられます。

次の作業

syslog フィルタ モジュールを作成した後、デバイスからアクセス可能な場所にファイルを保存する必要があります。ファイルは、ローカル システム メモリにコピーするか、ネットワーク ファイル サーバに保存できます。

Embedded Syslog Manager の設定

Embedded Syslog Manager（ESM）を設定するには、生成された syslog メッセージに適用する 1 つまたは複数のフィルタを指定し、syslog メッセージのターゲットを指定します。

はじめる前に

デバイスで 1 つまたは複数の syslog フィルタ モジュールを使用できなければなりません。

手順の概要

1. イネーブル化
2. **configureterminal**
3. **loggingfilter***filter-url* [*position*] [**args***filter-arguments*]
4. システム ロギング出力に適用する必要がある各 syslog フィルタ モジュールに対して、ステップ 3 を繰り返します。
5. 次のいずれか 1 つを入力します。
   - **logging** [**console** | **buffered** | **monitor**] **filtered** [*security-level*]
   - または
   - **logginghost** {*ip-address* | *hostname*} **filtered** [**stream***stream-id*]
6. 希望する各システム ロギング宛先に対して、ステップ 5 を繰り返します。
7. **loggingsource-interface***typenumber*
8. **loggingorigin-id** {**hostname** | **ip** | **ipv6** |**string***user-defined-id*}
9. **end**
10. **showlogging**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	イネーブル化 例： `Device> enable`	特権 EXEC モードをイネーブルにします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configureterminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**loggingfilter***filter-url* [*position*] [**args***filter-arguments*] 例： `Device(config)# logging filter slot0:/escalate.tcl 1 args CONFIG_I 1`	生成されたシステム ロギング メッセージに適用する 1 つまたは複数の syslog フィルタ モジュールを指定します。 • 使用する必要がある各 syslog フィルタ モジュールに対して、このコマンドを繰り返します。 • *filter-url* 引数は、syslog フィルタ モジュール（スクリプト）を示す Cisco IOS ファイル システムの場所です。場所としてローカル メモリまたはリモート サーバが可能です（**tftp:**、**ftp:**、または **rcp:**を使用）。

	コマンドまたはアクション	目的
		• オプションの *position* 引数は、syslog フィルタ モジュールを実行する順序を指定します。この引数を省略した場合、指定されたモジュールは、チェーンの最後のモジュールとして配置されます。 • 再び **loggingfilter** コマンドを入力して異なる位置（position）を指定することにより、フィルタの順序をすばやく再設定できます。 • オプションの **args***filter-arguments* 構文を追加すると、指定されたフィルタに引数を渡すことができます。複数の引数を指定できます。引数の数とタイプは、syslog フィルタ モジュールで定義する必要があります。たとえば、特定の E メール アドレスを引数として受け入れるように syslog フィルタ モジュールが設計されている場合、**argsuser@host.com** 構文を使用して E メール アドレスを渡すことができます。複数の引数は、通常、スペースで区切ります。 • 実行するモジュールのリストからモジュールを削除するには、このコマンドの **no** 形式を使用します。
ステップ 4	システム ロギング出力に適用する必要がある各 syslog フィルタ モジュールに対して、ステップ 3 を繰り返します。	--

ステップ 5	次のいずれか1つを入力します。
	• **logging** [**console** | **buffered** | **monitor**] **filtered** [*security-level*]
	• または
	• **logginghost** {*ip-address* | *hostname*} **filtered** [**stream***stream-id*]
	
	例：
	`Device(config)# logging console filtered informational`
	
	例：
	`Device(config)# logging host 209.165.200.225 filtered stream 20`
		ESM のフィルタリングされた syslog 出力のターゲットを指定します。
		• ESM フィルタリングされた syslog メッセージの送信先として、コンソール、モニタ（TTY および Telnet 接続）、システム バッファ、またはリモート ホストが可能です。
		• オプションの *level* 引数は、メッセージの送信を、指定された値またはそれより低い値のメッセージに限定します。たとえば、レベル **1** が指定されている場合、レベル 1（アラート）またはレベル 0（緊急事態）のメッセージだけが、指定されたターゲットに送信されます。レベルはキーワードまたは数字として指定できます。
		• コンソール、モニタ接続、またはシステム バッファにロギングする場合、*level* 引数によって指定された重大度のしきい値が ESM フィルタリングよりも優先されます。ESM ターゲットに送信されるべきメッセージが ESM フィルタから返された場合でも、重大度が設定済みしきい値を満たさない（つまりレベル値より高い）場合には、メッセージが送信されません。
		• リモート ホストにロギングする場合、ストリーム タグを使用すると、メッセージ タイプに基づいて宛先を指定できます。**stream***stream-id* 構文を使用すると、指定されたストリーム値を持つ

	コマンドまたはアクション	目的
		メッセージだけを特定のホストに送信するよう ESM を設定できます。 • ストリーム値は、設定された syslog フィルタ モジュールによって、メッセージに適用されます。たとえば、重大度 5 のすべてのメッセージでストリーム タグ「20」を設定できます。さらに、ストリーム タグ「20」を持つすべてのメッセージが 209.165.200.225 のホストに送信されるように指定できます。
ステップ 6	希望する各システム ロギング宛先に対して、ステップ 5 を繰り返します。	• ロギング ホスト コマンドを複数回発行することによって、異なるシステム ロギング ストリームに異なるターゲットを指定できます。 • コンソール、モニタ接続、またはシステム バッファに向けて、異なる重大度のメッセージを送信するように設定できます。たとえば、ネットワーク オペレーション センター（NOC）の画面に重要なメッセージだけを表示できます（モニタまたはコンソール接続を使用）。
ステップ 7	**loggingsource-interface***typenumber* 例： `Device(config)# logging source-interface Gigabitethernet 0/0`	（任意）リモート syslog ホストに送信する syslog メッセージの送信元インターフェイスを指定します。 • 通常では、リモート ホストに送信される syslog メッセージは、メッセージ生成の時点で使用できるあらゆるインターフェイスを使用します。このコマンドを使用すると、デバイスは、指定されたインターフェイスだけから syslog メッセージをリモート ホストに送信します。
ステップ 8	**loggingorigin-id** {**hostname** | **ip** | **ipv6** |**string***user-defined-id*} 例： `Device(config)# logging origin-id string “Domain 2, Router 5”`	（任意）リモート ホストに送信される syslog メッセージに発信元 ID を追加できます。 • 発信元 ID は、リモート ホストに送信されるすべての syslog メッセージの最初に追加されます。ID はホスト名、IP アドレス、または指定する任意のテキストです。 • 発信元 ID は、syslog 出力を複数のデバイスから 1 つの syslog ホストに送信する場合、システム ロギング メッセージの送信元を識別するのに役立ちます。
ステップ 9	**end** 例： `Device(config)# end`	現在のコンフィギュレーション セッションを終了し、CLI を特権 EXEC モードに返します。

	コマンドまたはアクション	目的
ステップ 10	**showlogging** 例： `Device# show logging`	（任意）ESMのフィルタリングされたロギングのステータスを含む、システム ロギングのステータスを表示します。 • バッファへのフィルタリングされたロギングがイネーブルの場合、このコマンドは、バッファに保存されたデータも表示します。 • syslog フィルタ モジュールをこのコマンドの出力に示す順番は、フィルタ モジュールが実行される順番です。

Embedded Syslog Manager の設定例

例：Embedded Syslog Manager の設定例

次の例では、コンソール接続の Embedded Syslog Manager（ESM）フィルタ ロギングがイネーブル、モニタ接続とバッファでの標準ロギングがイネーブル、ホスト 209.165.200.225 での XML 形式のロギングがイネーブルです。

```
Device(config)# logging filter tftp://209.165.200.225/ESM/escalate.tcl
Device(config)# logging filter slot0:/email.tcl user@example.com
Device(config)# logging filter slot0:/email_guts.tcl
Device(config)# logging console filtered
Device(config)# logging monitor 4
Device(config)# logging buffered debugging
Device(config)# logging host 209.165.200.225 xml
Device(config)# end

Device# show logging
Syslog logging: enabled (0 messages dropped, 8 messages rate-limited,
               0 flushes, 0 overruns, xml disabled, filtering enabled)
    Console logging: level debugging, 21 messages logged, xml disabled,
                     filtering enabled
    Monitor logging: level warnings , 0 messages logged, xml disabled,
                     filtering disabled
    Buffer logging: level debugging, 30 messages logged, xml disabled,
                     filtering disabled
    Logging Exception size (8192 bytes)
    Count and timestamp logging messages: disabled

Filter modules:
    tftp://209.165.200.225/ESM/escalate.tcl
    slot0:/email.tcl user@example.com

    Trap logging: level informational, 0 message lines logged
        Logging to 209.165.200.225, 0 message lines logged, xml enabled,
              filtering disabled

Log Buffer (8192 bytes):

*Jan 24 09:34:28.431: %SYS-5-CONFIG_I: Configured from console by console
*Jan 24 09:34:51.555: %SYS-5-CONFIG_I: Configured from console by console
*Jan 24 09:49:44.295: %SYS-5-CONFIG_I: Configured from console by console
```

```
Device#
```

例：syslog フィルタ モジュール

syslog スクリプト モジュールは Tcl スクリプトです。独自の syslog スクリプト モジュールの開発に役立つよう、次の例を示します。

（注） これらのスクリプト モジュールは単に例として提供されており、シスコによるサポートの対象外です。これらのスクリプトの機能または影響に関して、明示的または黙示的に保証することはありません。

例：重大度のエスカレーション

この ESM syslog フィルタ モジュール例では、1 つのニーモニック（最初の CLI 引数経由で供給される）を監視し、メッセージの重大度を 2 つめの CLI 引数によって指定される重大度に拡大します。

```
#  ====================================================================
#  Embedded Syslog Manager                            ||        ||
#                                                     ||        ||
#  Severity Escalation Filter                        ||||      ||||
#                                              ..:||||||:..:||||||:..
#                                             ------------------------
#                                             C i s c o  S y s t e m s
#  ====================================================================
#
# Usage: Set CLI Args to "mnemonic new_severity"
#
# Namespace: global
# Check for null message
if { [string length $::orig_msg] == 0} {
   return ""
}

if { [info exists ::cli_args] } {
    set args [split $::cli_args]
    if { [ string compare -nocase [lindex $args 0] $::mnemonic ] == 0 } {
        set ::severity [lindex $args 1]
        set sev_index [ string first [lindex $args 0] $::orig_msg ]
        if {  $sev_index >= 2 } {
           incr sev_index -2
           return [string replace $::orig_msg $sev_index $sev_index \
              [lindex $args 1]]
        }
    }
}
return $::orig_msg
```

例：メッセージのカウント

この ESM syslog フィルタ モジュール例は、読みやすくするために 2 つのファイルに分割されています。最初のファイルで、ユーザは msg_to_watch アレイを設定することにより、カウント対象のメッセージおよび要約する頻度（相関期間）を設定できます。実際の手順は counting_guts.tcl ファ

イルに含まれています。バックグラウンド処理を行う可能性がある他のESMフィルタとの競合を避けるために、別個の名前空間「counting」を使用していることに注意してください。

```
#  ==================================================================
#  Embedded Syslog Manager                          ||        ||
#                                                   ||        ||
#  Message Counting Filter                         ||||      ||||
#                                             ..:||||||:..:||||||:..
#                                            ------------------------
#                                            C i s c o  S y s t e m s
# ==========================================================================


#
# Usage:
# 1) Define the location for the counting_guts.tcl script
#
# 2) Define message categories to count and how often to dump them (sec)
#    by populating the "msg_to_watch" array below.
#    Here we define category as facility-severity-mnemonic
#    Change dump time to 0 to disable counting for that category
#
# Namespace: counting
namespace eval ::counting {
    set sub_script_url tftp://172.16.0.0/12/ESM/counting_guts.tcl
    array set msg_to_watch {
        SYS-5-CONFIG_I            5
    }
#  ======================= End User Setup ==============================
# Initialize processes for counting
    if { [info exists init] == 0 } {
       source $sub_script_url
       set position $module_position
    }
# Process the message
process_category
} ;# end namespace counting
```

メッセージ カウント サポート モジュール（counting_guts.tcl）

```
#  ==================================================================
#  Embedded Syslog Manager                          ||        ||
#                                                   ||        ||
#  Message Counting Support Module                 ||||      ||||
#                                             ..:||||||:..:||||||:..
#  (No User Modification)                     ------------------------
#                                             C i s c o  S y s t e m s
# ==========================================================================



namespace eval ::counting {

# namespace variables

array set cat_msg_sev {}
array set cat_msg_traceback {}
array set cat_msg_pid {}
array set cat_msg_proc {}
array set cat_msg_ts {}
array set cat_msg_buginfseq {}
array set cat_msg_name {}
array set cat_msg_fac {}
array set cat_msg_format {}
array set cat_msg_args {}
array set cat_msg_count {}
array set cat_msg_dump_ts {}
```

```
# Should I count this message ?
   proc query_category {cat} {
        variable msg_to_watch
        if { [info exists msg_to_watch($cat)] } {
            return $msg_to_watch($cat)
        } else {
            return 0
        }
    }
   proc clear_category {index} {
        variable cat_msg_sev
        variable cat_msg_traceback
        variable cat_msg_pid
        variable cat_msg_proc
        variable cat_msg_ts
        variable cat_msg_buginfseq
        variable cat_msg_name
        variable cat_msg_fac
        variable cat_msg_format
        variable cat_msg_args
        variable cat_msg_count
        variable cat_msg_dump_ts
        unset cat_msg_sev($index) cat_msg_traceback($index) cat_msg_pid($index)\
              cat_msg_proc($index) cat_msg_ts($index) \
              cat_msg_buginfseq($index)  cat_msg_name($index) \
              cat_msg_fac($index) cat_msg_format($index) cat_msg_args($index)\
              cat_msg_count($index) cat_msg_dump_ts($index)
    }
# send out the counted messages
   proc dump_category {category} {
        variable cat_msg_sev
        variable cat_msg_traceback
        variable cat_msg_pid
        variable cat_msg_proc
        variable cat_msg_ts
        variable cat_msg_buginfseq
        variable cat_msg_name
        variable cat_msg_fac
        variable cat_msg_format
        variable cat_msg_args
        variable cat_msg_count
        variable cat_msg_dump_ts
        variable poll_interval
        set dump_timestamp [cisco_service_timestamp]
foreach index [array names cat_msg_count $category] {
      set fsm "$cat_msg_fac($index)-$cat_msg_sev($index)-$cat_msg_name($index)"
      set ::orig_msg \
        [format "%s%s: %%%s: %s %s %s %s - (%d occurence(s) between %s and %s)"\
        $cat_msg_buginfseq($index)\
          $dump_timestamp\
          $fsm \
          [uplevel 1 [linsert $cat_msg_args($index) 0 ::format
$cat_msg_format($index) ]] \
          $cat_msg_pid($index) \
          $cat_msg_proc($index) \
          $cat_msg_traceback($index) \
          $cat_msg_count($index) \
          $cat_msg_ts($index) \
          $dump_timestamp]
# Prepare for remaining ESM filters
                set ::severity $cat_msg_sev($index)
                set ::traceback $cat_msg_traceback($index)
                set ::pid $cat_msg_pid($index)
                set ::process $cat_msg_proc($index)
                set ::timestamp $cat_msg_ts($index)
                set ::buginfseq $cat_msg_buginfseq($index)
                set ::mnemonic $cat_msg_name($index)
                set ::facility $cat_msg_fac($index)
                set ::format_string $cat_msg_format($index)
                set ::msg_args [split $cat_msg_args($index)]
                esm_errmsg $counting::position
                clear_category $index
        }
```

```
  }
# See if this message already has come through since the last dump.
# If so, increment the count, otherwise store it.
   proc process_category {} {
        variable cat_msg_sev
        variable cat_msg_traceback
        variable cat_msg_pid
        variable cat_msg_proc
        variable cat_msg_ts
        variable cat_msg_buginfseq
        variable cat_msg_name
        variable cat_msg_fac
        variable cat_msg_format
        variable cat_msg_args
        variable cat_msg_count
        variable cat_msg_dump_ts
        if { [string length $::orig_msg] == 0} {
           return ""
        }
        set category "$::facility-$::severity-$::mnemonic"
        set correlation_window [expr [ query_category $category ] * 1000]
        if { $correlation_window == 0 } {
            return $::orig_msg
        }
        set message_args [join $::msg_args]
        set index "$category,[lindex $::msg_args 0]"
        if { [info exists cat_msg_count($index)] } {
                 incr cat_msg_count($index)
         } else {
              set cat_msg_sev($index) $::severity
              set cat_msg_traceback($index) $::traceback
              set cat_msg_pid($index) $::pid
              set cat_msg_proc($index) $::process
              set cat_msg_ts($index) $::timestamp
              set cat_msg_buginfseq($index) $::buginfseq
              set cat_msg_name($index) $::mnemonic
              set cat_msg_fac($index) $::facility
              set cat_msg_format($index) $::format_string
              set cat_msg_args($index) $message_args
              set cat_msg_count($index) 1
              set cat_msg_dump_ts($index) [clock seconds]
              catch [after $correlation_window counting::dump_category $index]
         }
         return ""
   }
# Initialized
set init 1
} ;#end namespace counting
```

例：XML タギング

この ESM syslog フィルタ モジュールは、ユーザ定義の XML タグを syslog メッセージに適用します。

```
#  ==================================================================
#  Embedded Syslog Manager                          ||        ||
#                                                   ||        ||
#  XML Tagging Filter                              ||||      ||||
#                                             ..:||||||:..:||||||:..
#                                            ------------------------
#                                            C i s c o  S y s t e m s
#  ==================================================================
#
# Usage: Define desired tags below.
#
# Namespace: xml
# Check for null message
    if { [string length $::orig_msg] == 0} {
       return ""
```

```
    }
namespace eval xml {
####  define tags ####
set MSG_OPEN "<ios-log-msg>"
set MSG_CLOSE "</ios-log-msg>"
set FAC_OPEN   "<facility>"
set FAC_CLOSE  "</facility>"
set SEV_OPEN   "<severity>"
set SEV_CLOSE  "</severity>"
set MNE_OPEN   "<msg-id>"
set MNE_CLOSE  "</msg-id>"
set SEQ_OPEN   "<seq>"
set SEQ_CLOSE  "</seq>"
set TIME_OPEN  "<time>"
set TIME_CLOSE "</time>"
set ARGS_OPEN  "<args>"
set ARGS_CLOSE "</args>"
set ARG_ID_OPEN "<arg id="
set ARG_ID_CLOSE "</arg>"
set PROC_OPEN "<proc>"
set PROC_CLOSE "</proc>"
set PID_OPEN "<pid>"
set PID_CLOSE "</pid>"
set TRACE_OPEN "<trace>"
set TRACE_CLOSE "</trace>"
#  ======================= End User Setup ==============================
#### clear result ####
set result ""
#### message opening, facility, severity, and name ####
append result $MSG_OPEN $FAC_OPEN $::facility $FAC_CLOSE $SEV_OPEN $::severity
$SEV_CLOSE $MNE_OPEN $::mnemonic $MNE_CLOSE
#### buginf sequence numbers ####
if { [string length $::buginfseq ] > 0 } {
    append result $SEQ_OPEN $::buginfseq $SEQ_CLOSE
}
#### timestamps ####
if { [string length $::timestamp ] > 0 } {
    append result $TIME_OPEN $::timestamp $TIME_CLOSE
}
#### message args ####
if { [info exists ::msg_args] } {
    if { [llength ::msg_args] > 0 } {
        set i 0
        append result $ARGS_OPEN
        foreach arg $::msg_args {
            append result $ARG_ID_OPEN $i ">" $arg $ARG_ID_CLOSE
                incr i
        }
        append result $ARGS_CLOSE
    }
}
#### traceback ####
if { [string length $::traceback ] > 0 } {
    append result $TRACE_OPEN $::traceback $TRACE_CLOSE
}
#### process ####
if { [string length $::process ] > 0 } {
    append result $PROC_OPEN $::process $PROC_CLOSE
}
#### pid ####
if { [string length $::pid ] > 0 } {
    append result $PID_OPEN $::pid $PID_CLOSE
}
#### message close ####
append result $MSG_CLOSE
return "$result"
} ;# end namespace xml
```

例：SMTP ベースの電子メール アラート

この ESM syslog フィルタ モジュール例では、コンフィギュレーション メッセージを監視し、CLI 引数として供給される E メール アドレスに送信します。このフィルタは 2 つのファイルに分割されています。最初のファイルはフィルタを実装し、2 番目のファイルは Simple Mail Transfer Protocol（SMTP）クライアントを実装します。

```
#  ==================================================================
#  Embedded Syslog Manager                          ||        ||
#                                                   ||        ||
#  Email Filter                                    ||||      ||||
#  (Configuration Change Warning)               ..:||||||:..:||||||:..
#                                             ------------------------
#                                              C i s c o  S y s t e m s
#   ================================================================
# Usage:  Provide email address as CLI argument.  Set email server IP in
#         email_guts.tcl
#
# Namespace: email
if { [info exists email::init] == 0 } {
   source tftp://123.123.123.123/ESM/email_guts.tcl
}
# Check for null message
if { [string length $::orig_msg] == 0} {
      return ""
   }
if { [info exists ::msg_args] } {
    if { [string compare -nocase CONFIG_I $::mnemonic ] == 0 } {
                email::sendmessage $::cli_args $::mnemonic \
                [string trim $::orig_msg]
    }
}
return $::orig_msg
```

E メール サポート モジュール（email_guts.tcl）

```
#  ==================================================================
#  Embedded Syslog Manager                          ||        ||
#                                                   ||        ||
#  Email Support Module                            ||||      ||||
#                                               ..:||||||:..:||||||:..
#                                             ------------------------
#                                              C i s c o  S y s t e m s
#   ================================================================
#
# Usage: Set email host IP, from, and friendly strings below.
#
namespace eval email {
    set sendmail(smtphost)172.16.0.1
    set sendmail(from) $::hostname
    set sendmail(friendly) $::hostname
    proc sendmessage {toList subject body} {
        variable sendmail
        set smtphost $sendmail(smtphost)
        set from $sendmail(from)
        set friendly $sendmail(friendly)
        set sockid [socket $smtphost 25]
## DEBUG
set status [catch {
        puts $sockid "HELO $smtphost"
        flush $sockid
        set result [gets $sockid]
        puts $sockid "MAIL From:<$from>"
        flush $sockid
        set result [gets $sockid]
        foreach to $toList {
```

```
            puts $sockid "RCPT To:<$to>"
            flush $sockid
        }
        set result [gets $sockid]
        puts  $sockid "DATA "
        flush $sockid
        set result [gets  $sockid]
        puts  $sockid "From: $friendly <$from>"
        foreach to $toList {
            puts $sockid "To:<$to>"
        }
        puts  $sockid "Subject: $subject"
        puts  $sockid "\n"
        foreach line [split $body "\n"] {
            puts  $sockid " $line"
        }
        puts  $sockid "."
        puts  $sockid "QUIT"
        flush $sockid
        set result [gets  $sockid]
} result]
        catch {close $sockid }
    if {$status} then {
        return -code error $result
    }
}
} ;# end namespace email
set email::init 1
```

例：ストリーム

この ESM syslog フィルタ モジュールの例では、特定のファシリティを監視し（最初の CLI 引数）、これらのメッセージを特定のストリームにルーティングします（2 番目の CLI 引数）。

```
#  ==================================================================
#  Embedded Syslog Manager                          ||        ||
#                                                   ||        ||
#  Stream Filter (Facility)                        ||||      ||||
#                                             ..:||||||:..:||||||:..
#                                            ------------------------
#                                             C i s c o  S y s t e m s
#   ==================================================================
# Usage:  Provide facility and stream as CLI arguments.
#
# Namespace: global
# Check for null message
#  ======================= End User Setup ==============================
set args [split $::cli_args]
if { [info exists ::msg_args] } {
    if { $::facility == [lindex $args 0] } {
                set ::stream [lindex $args 1]
    }
}
return $::orig_msg}
```

例：送信元 IP タギング

loggingsource-interface CLI コマンドを使用すると、デバイスから送信されるすべての syslog パケットで送信元 IP アドレスを指定できます。次の syslog フィルタ モジュールの例では、フィルタ モジュール内で **show** CLI コマンド（この場合は **showrunning-config** および **showipinterface**）を使用して、syslog メッセージに送信元 IP アドレスを追加する方法を示します。スクリプトは、最初に「source_ip::init」というローカル名前空間変数を検索します。処理される最初の syslog メッセー

ジで変数が定義されていない場合、フィルタは**show** コマンドを実行し、正規表現を使用して、送信元インターフェイスとその IP アドレスを取得します。

このスクリプトでは、**show** コマンドが 1 回だけ実行されることに注意してください。送信元インターフェイスまたはその IP アドレスが変更される場合、新しい情報を取得するためにフィルタを再び初期化する必要があります（show コマンドをすべての syslog メッセージで実行できますが、これはうまく拡張できません）。

```
#  ==================================================================
#  Embedded Syslog Manager                                ||        ||
#                                                         ||        ||
#  Source IP Module                                      ||||      ||||
#                                                  ..:||||||:..:||||||:..
#                                                ------------------------
#                                                 C i s c o  S y s t e m s
#   ==================================================================
# Usage: Adds Logging Source Interface IP address to all messages.
#
# Namespace:source_ip
#
#  ======================= End User Setup ==============================
namespace eval ::source_ip {
    if { [info exists init] == 0 } {
       if { [catch {regexp {^logging source-interface (.*$)} [exec show
run | inc logging source-interface] match source_int}]} {
           set suffix "No source interface specified"
       } elseif { [catch {regexp {Internet address is (.*)/.*$} [exec
show ip int $source_int | inc Internet] match ip_addr}]} {
           set suffix "No IP address configured for source interface"
       } else {
           set suffix $ip_addr
       }
       set init 1
    }

    if { [string length $::orig_msg] == 0} {
      return ""
    }
    return "$::orig_msg - $suffix"
} ;# end namespace source_ip
```

Embedded Syslog Manager に関する追加情報

関連資料

関連項目	マニュアル タイトル
Cisco IOS コマンド	『Cisco IOS Master Commands List, All Releases』
Cisco IOS XE コマンド	『Command Lookup Tool』
システム メッセージ ロギング	「Troubleshooting and Fault Management」モジュール
XML 形式のシステム メッセージ ロギング	「XML Interface to Syslog Messages」モジュール
シスコ ソフトウェアでの Tcl 8.3.4 サポート	「Cisco IOS Scripting with Tcl」モジュール

関連項目	マニュアル タイトル
ネットワーク管理コマンド（logging コマンドを含む）：コマンド構文の詳細、デフォルト設定、コマンドモード、コマンド履歴、使用上のガイドライン、および例	『*Cisco IOS Network Management Command Reference*』

標準規格および RFC

規格/RFC	タイトル
新しい規格または変更された規格はサポートされていません。また、既存の規格に対するサポートに変更はありません。	--
RFC-3164	『*The BSD Syslog Protocol*』 この RFC は、情報提供のために記載しています。シスコによる syslog の実装では、この RFC で言及されているプロトコルガイドラインとの完全な準拠性を要求していません。 サポートされている RFC がすべて記載されているわけではありません。

MIB

MIB	MIB のリンク
新しい規格または変更された規格はサポートされていません。また、既存の規格に対するサポートに変更はありません。	選択したプラットフォーム、Cisco IOS リリース、およびフィーチャ セットに関する MIB を探してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

シスコのテクニカル サポート

説明	リンク
★枠で囲まれた Technical Assistance の場合★右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

Embedded Syslog Manager の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

表 *1*：*Embedded Syslog Manager* の機能情報

機能名	リリース	機能情報
Embedded Syslog Manager		Embedded Syslog Manager（ESM）機能は、Cisco IOS システム メッセージ ロガーによる伝送前に、システム ロギング メッセージのフィルタリング、拡大、相互の関連付け、ルーティング、およびカスタマイズを可能にするプログラマブル フレームワークを提供します。

用語集

（注） この用語集に記載されていない用語については、『Internetworking Terms and Acronyms』を参照してください。

コンソール：デバイスのコンソールポートへの接続（CTYまたはコンソール回線）を示します。一般的にこれはコンソールポートに直接接続された端末または端末エミュレーションプログラムを備えたPCです。**showterminal** コマンドに対応します。

モニタ：ライン ポートでのTTY（TeleTYpe端末）ライン接続を示します。つまり、「モニタ」キーワードは、端末回線接続またはTelnet（端末エミュレーション）接続に対応します。TTYライン（ポートとも呼ばれます）は、端末、モデム、シリアル プリンタなどの周辺装置と通信します。TTY接続の例として、ダイヤルアップ モデムを使用してデバイスに接続する端末エミュレーション プログラムを備えたPCがあります。

SEM：システム エラー メッセージの略。システムロギング（syslog）プロセスによって生成されるメッセージを指す用語として、「システムエラーメッセージ」が以前に使用されていました。syslogメッセージは標準化された形式を使用し、「緊急」（レベル0）から「デバッグ」（レベル7）までの8つの重大度があります。これらのメッセージには、エラー以外のデバイス動作の通知（情報の通知など）が含まれる場合もあるため、「システムエラーメッセージ」という用語は、実際に誤解を招くおそれがあります。

syslog：シスコ ソフトウェアでのシステム メッセージ ロギング プロセスの略。また、「syslogメッセージ」のように、生成されたメッセージを指す場合もあります。専門用語としての「syslog」はリモート ホストへのメッセージ ロギング プロセスのみを指しますが、一般的には、すべてのシスコ システム ロギング プロセスを示すために使用されます。

トラップ：エラーメッセージを送信するためのシステムソフトウェア内のトリガー。「トラップロギング」とは、リモートホストへのメッセージのロギングを意味します。リモートホストはトラップ メッセージを送信するデバイスから見ると実際にはsyslogホストですが、受信側デバイスは、収集されたsyslogデータを他のデバイスに提供することが多いため、受信側デバイスもまた「syslogサーバ」と呼ばれます。

第 3 章

ローカル不揮発性ストレージへのロギング

ローカル不揮発性ストレージへのロギング機能では、システム ロギング メッセージを Advanced Technology Attachment フラッシュ ディスクに保存できます。デバイスが再起動しても、ブートフラッシュまたはハードディスクに保存されたメッセージは消去されません。

- 機能情報の確認，27 ページ
- ローカル不揮発性ストレージへのロギングの前提条件，28 ページ
- ローカル不揮発性ストレージへのロギングの制約事項，28 ページ
- ローカル不揮発性ストレージへのロギングに関する情報，28 ページ
- ローカル不揮発性ストレージへのロギングの設定方法，29 ページ
- ローカル不揮発性ストレージへのロギングの設定例，31 ページ
- その他の参考資料，31 ページ
- ローカル不揮発性ストレージへのロギングの機能情報，32 ページ

機能情報の確認

ご使用のソフトウェア リリースでは、このモジュールで説明されるすべての機能がサポートされているとは限りません。最新の機能情報および警告については、Bug Search Tool およびご使用のプラットフォームおよびソフトウェア リリースのリリース ノートを参照してください。このモジュールに記載されている機能の詳細を検索し、各機能がサポートされているリリースのリストを確認する場合は、このモジュールの最後にある機能情報の表を参照してください。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

ローカル不揮発性ストレージへのロギングの前提条件

logging buffered コマンドをイネーブルにする

loggingpersistent コマンドを使用して、ローカル不揮発性ストレージへのロギング機能をイネーブルにする前に、**loggingbuffered** コマンドを使用して、内部バッファへのメッセージのロギングをイネーブルにする必要があります。詳細については、「ブートフラッシュまたはハードディスクへのロギング メッセージの書き込み」セクションを参照してください。

ローカル不揮発性ストレージへのロギングの制約事項

使用できるブートフラッシュまたはハードディスクの容量によって、保存されるログ ファイルのサイズと数が制限される

システム ロギング メッセージに割り当てられるブートフラッシュまたはハードディスクの容量によって、保存できるロギング ファイルの数が制限されます。割り当てしきい値を超えると、ディレクトリ内の最も古いログ ファイルが削除され、新しいシステム ロギング メッセージ用の容量を用意します。システム ロギング メッセージを恒久的に保存するには、外部デバイスにアーカイブする必要があります。詳細については、「外部ディスクへのロギング メッセージのコピー」セクションを参照してください。

（注） ローカル不揮発性ストレージへのロギングは、最大2GBの記憶域を使用する場合があります。

ローカル不揮発性ストレージへのロギングに関する情報

システム ロギング メッセージ

システム ロギング メッセージには、デバイスのアプリケーション プログラミング インターフェイス（API）によって生成されたエラーおよびデバッグ メッセージが含まれます。一般的に、ロギング メッセージはデバイスのメモリ バッファに保存され、バッファが満杯になると古いメッセージが新しいメッセージで上書きされます。デバイスが再起動すると、すべてのロギング メッセージがメモリ バッファから消去されます。

ローカル不揮発性ストレージへのロギングの設定方法

ブートフラッシュまたはハードディスクへのロギングメッセージの書き込み

ローカル不揮発性ストレージへのロギング機能をイネーブルにし、ブートフラッシュまたはハードディスクにロギング メッセージを書き込むには、次の作業を実行します。

手順の概要

1. イネーブル化
2. **configureterminal**
3. **loggingbuffered** [*buffer-size* | *severity-level*]
4. **loggingpersistent** [**urlharddisk:**/*directory*] [**size***filesystem-size*] [**filesize***logging-file-size*]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	イネーブル化 例： `Device> enable`	特権 EXEC モードをイネーブルにします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configureterminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードをイネーブルにします。
ステップ 3	**loggingbuffered** [*buffer-size* | *severity-level*] 例： `Device(config)# logging buffered`	ローカル バッファへのシステム メッセージ ロギングをイネーブルにし、バッファにロギングするメッセージを重大度に基づいて制限します。 • オプションの *buffer-size* の引数は、バッファのサイズを指定します。指定できる値の範囲は 4096 ～ 4294967295 です。デフォルトのサイズは、プラットフォームによって異なります。 • オプションの *severity-level* 引数は、バッファへのメッセージのロギングを、指定された重大度またはそれ以上のレベルのメッセージに制限します。

	コマンドまたはアクション	目的
ステップ 4	**loggingpersistent** [**urlharddisk:**/*directory*] [**size***filesystem-size*] [**filesize***logging-file-size*] 例： `Device(config)# logging persistent url harddisk:/syslog size 134217728 filesize 16384` （注） デフォルト値は次のとおりです。url: bootflash:/syslog filesystem-size: 10% of total disk space logging-file-size: 262144	メモリ バッファ内のロギング メッセージを、デバイスのブートフラッシュまたはハードディスク上の指定のディレクトリに書き込みます。 • ロギング メッセージをブートフラッシュまたはハードディスク上のファイルに書き込む前に、シスコ ソフトウェアは、十分なディスク領域があるかどうかをチェックします。十分なディスク スペースがない場合、ロギング メッセージの最も古いファイル（タイムスタンプによる）が削除され、現在のファイルが保存されます。 • ログ ファイルのファイル名フォーマットは log_MM:DD:YYYY::hh:mm:ss です（例：log_11:26:2012::01:01:41）。 （注） この機能は、そのファイル名の形式（秒レベルまでのタイムスタンプ サフィックスが含まれている）により、1 秒あたり 1 つのログ ファイルだけをサポートします。 （注） このコマンドのデフォルトは次のとおりです。 • **url: bootflash:/syslog** Filesystem-size: 10% of total disk space.Logging-file-size: 262144

外部ディスクへのロギング メッセージのコピー

ロギング メッセージを、ブートフラッシュまたはハードディスクから外部ディスクにコピーするには、次の作業を実行します。

手順の概要

1. イネーブル化
2. **copy***source-urldestination-url*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	イネーブル化 例： `Device> enable`	特権 EXEC モードをイネーブルにします。 • パスワードを入力します（要求された場合）。

	コマンドまたはアクション	目的
ステップ 2	**copy***source-urldestination-url* 例： `Device# copy harddisk:/syslog ftp://myuser/mypass@192.168.1.129/syslog`	ブートフラッシュまたはハードディスク上の指定のファイルまたはディレクトリを、FTP 経由で指定の URL にコピーします。

ローカル不揮発性ストレージへのロギングの設定例

例：ブートフラッシュまたはハードディスクへのロギングメッセージの書き込み

次に、最大 134217728 バイト（128 MB）のロギング メッセージをディスク 0 の syslog ディレクトリに書き込み、16384 バイトのファイル サイズを指定する例を示します。

```
Device(config)# logging buffered
Device(config)# logging persistent url harddisk:/syslog size 134217728 filesize 16384
```

例：外部ディスクへのロギング メッセージのコピー

次に、ロギング メッセージをデバイスのブートフラッシュまたはハードディスクから外部ディスクにコピーする例を示します。

```
Device# copy harddisk:/syslog ftp://myuser/mypass@192.168.1.129/syslog
```

その他の参考資料

関連資料

関連項目	マニュアル タイトル
copy コマンド	『Cisco IOS Configuration Fundamentals Command Reference』

関連項目	マニュアル タイトル
ネットワーク管理コマンド（loggingコマンドを含む）：コマンド構文の詳細、デフォルト設定、コマンドモード、コマンド履歴、使用上のガイドライン、および例	『Cisco IOS Network Management Command Reference』

MIB

MIB	MIB のリンク
• この機能によってサポートされる新しいMIBまたは変更されたMIBはありません。またこの機能による既存 MIB のサポートに変更はありません。	選択したプラットフォーム、Cisco IOS XE Release、およびフィーチャ セットの MIB を検索してダウンロードするには、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

シスコのテクニカル サポート

説明	リンク
★枠で囲まれた Technical Assistance の場合★右の URL にアクセスして、シスコのテクニカルサポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

ローカル不揮発性ストレージへのロギングの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

表 *2*：ローカル不揮発性ストレージへのロギングの機能情報

機能名	リリース	機能情報
ローカル不揮発性ストレージへのロギング	Cisco IOS XE Release 2.1	ローカル不揮発性ストレージへのロギング機能では、システム ロギング メッセージを Advanced Technology Attachment フラッシュ ディスクに保存できます。デバイスが再起動しても、ブートフラッシュまたはハードディスクに保存されたメッセージは消去されません。 導入または変更されたコマンドは、**loggingpersistent** です。

第 4 章

syslog の信頼性の高い伝送およびフィルタリング

syslog の信頼性の高い伝送およびフィルタリング機能によって、デバイスを syslog メッセージの受信用にカスタマイズできます。この機能は、ブロック拡張可能交換プロトコル（BEEP）を使用した syslog メッセージの信頼性の高いセキュアな伝送を提供します。さらに、基盤となる転送方式にかかわらず、1 つのロギング ホストへの複数のセッションを可能にし、メッセージ ディスクリミネータと呼ばれるフィルタリング メカニズムを提供します。

この章では、syslog 機能のための信頼性の高い伝送およびフィルタリングの機能と、それらのネットワーク内での設定方法について説明します。

- 機能情報の確認，35 ページ
- syslog の信頼性の高い伝送およびフィルタリングの前提条件，36 ページ
- syslog の信頼性の高い伝送およびフィルタリングの制約事項，36 ページ
- syslog の信頼性の高い伝送およびフィルタリングに関する情報，37 ページ
- syslog の信頼性の高い伝送およびフィルタリングの設定方法，43 ページ
- syslog の信頼性の高い伝送およびフィルタリングの設定例，50 ページ
- syslog トランザクションの VRF 対応送信元インターフェイスに関する追加情報，50 ページ
- syslog の信頼性の高い伝送およびフィルタリングの機能情報，51 ページ

機能情報の確認

ご使用のソフトウェア リリースでは、このモジュールで説明されるすべての機能がサポートされているとは限りません。最新の機能情報および警告については、Bug Search Tool およびご使用のプラットフォームおよびソフトウェア リリースのリリース ノートを参照してください。このモジュールに記載されている機能の詳細を検索し、各機能がサポートされているリリースのリストを確認する場合は、このモジュールの最後にある機能情報の表を参照してください。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

syslog の信頼性の高い伝送およびフィルタリングの前提条件

- デバイス レベルのレート制限を、ビジネス要件、ネットワーク トラフィック要件、またはパフォーマンス要件を満たすように設定します。
- 各 BEEP セッションには、RFC 3195 準拠の syslog-RAW 交換プロファイルが含まれている必要があります。
- 暗号イメージを使用する場合、プロビジョニング サービスに「DIGEST-MD5」を指定する Simple Authentication and Security Layer（SASL）プロファイルを確立する必要があります。
- syslog サーバは BEEP と互換性がある必要があります。
- syslog の信頼性の高い伝送およびフィルタリング機能の複数セッション機能を使用するには、syslog サーバ アプリケーションは、複数のセッションを処理できる必要があります。

syslog の信頼性の高い伝送およびフィルタリングの制約事項

- syslog-RAW、SASL、およびトランスポート層セキュリティ（TLS）プロファイルだけがサポートされています。
- syslog セッションの両端で同じ転送方式を使用する必要があります。
- メッセージディスクリミネータを特定の syslog セッションに関連付けるには、事前に定義する必要があります。
- syslog セッションは、1 つのメッセージ ディスクリミネータとだけ関連付けることができます。
- ユーザ データグラム プロトコル（UDP）によるメッセージ伝送は、TCP または BEEP による伝送よりも速くなります。

syslog の信頼性の高い伝送およびフィルタリングに関する情報

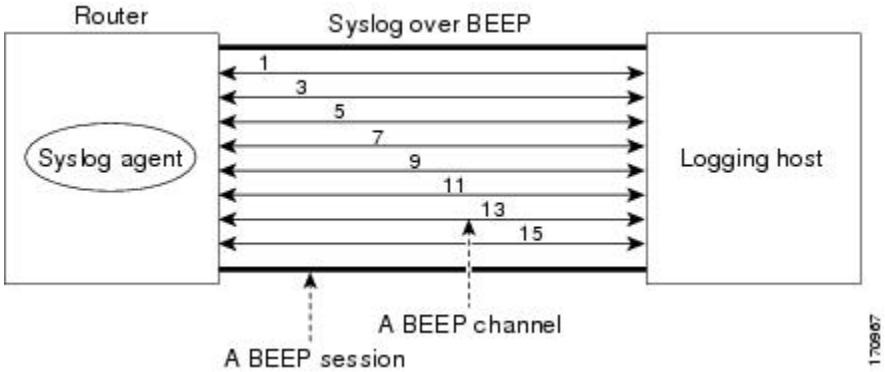
BEEP 転送のサポート

BEEPは、コネクション型非同期相互作用のための汎用アプリケーションプロトコルフレームワークです。これは、従来さまざまなプロトコルの実装で何度も利用されてきた機能を提供することを目的としています。BEEP は一般的に TCP 上で動作し、メッセージの交換が可能です。HTTP および同様のプロトコルとは異なり、接続の両端でいつでもメッセージを送信できます。BEEPには暗号化と認証のファシリティも含まれており、高い拡張性があります。

syslog メッセージの転送プロトコルとしての BEEP は、複数のチャネルを提供します。各チャネルは、同じホストへの異なるセッションに設定できます。BEEP は信頼性の高い転送を提供します。BEEP 接続を介して送信される syslog メッセージは、順序どおりに伝送されることが保証されます。

syslog の信頼性の高い伝送およびフィルタリング機能に導入されたコマンドライン インターフェイス（CLI）によって、新しい BEEP セッションが最大 8 つのチャネルを持つように設定できます。

次の図は、8 つの異なる syslog セッションを実現する、8 つのチャネルによる BEEP セッションを示しています。



チャネルは 1、3、5、7、9、11、13、および 15 と識別されます。使用できるチャネルの数（8）は、従来の RFC-3164 syslog メッセージ（0 ～ 7）の重大度の番号に対応して設計されています。メッセージ ディスクリミネータは、重大度が BEEP チャネルにマッピングされるように使用できます。インテリジェントな BEEP syslog サーバ（使用する BEEP スタックによって異なる）は、このマッピングを使用して、重大度の高いメッセージを優先できます（RFC 3081、セクション 3.1.4 を参照）。メッセージ ディスクリミネータと関連付けられている場合を除いて、すべての syslog セッション（チャネル）は、すべての syslog メッセージを受信します。

syslog メッセージ

syslog メッセージには、ホストが番号をメッセージの識別子として使用し、受信されたメッセージにギャップがあるかどうか検出できるようにするシーケンス番号があります。syslog メッセージには連続した番号が付けられます。BEEP の信頼性は、シーケンス番号（次の理由によって必要である）の必要性に代わるものではありません。

- シーケンス番号は、syslog メッセージを識別する簡単な方法を提供します。信頼性に関する問題に関係なく、シーケンス番号はメッセージ識別子として機能します。
- BEEP セッションは、syslog メッセージを送信するデバイスがアップ状態の間ずっと機能しているとは限りません。シーケンス番号は、BEEP セッションの間にメッセージが失われたかどうかを管理アプリケーションが評価する方法を提供します。
- BEEP はいくつかの転送のうちの 1 つにすぎません。信頼性の低い転送も使用されるので、syslog プロトコルは、常に提供されている信頼性の高い転送に依存すべきではありません。

syslog メッセージの既存の番号付け方式は、高度なメッセージの識別機能および複数のホストに対応するために、syslog の拡張で制限されます。メッセージの識別によって、シーケンス番号にギャップが発生します。つまり、ホストは、メッセージを見逃したかどうかを検出する能力を失います。シーケンス番号にギャップが発生しないよう、各セッションで syslog メッセージに連番が付けられた場合は、シーケンス番号でメッセージが一意に識別されなくなるため、どのメッセージが同じで、どのメッセージが異なるかを簡単に関連付けできなくなります。

識別をシーケンスおよび信頼性から分離するために、syslog メッセージに対して、次の変更が行われました。

- シーケンス番号は、メッセージの識別子として保持されます。低い番号を持つメッセージは、高い番号を持つメッセージよりも優先されますが、連続するように保証されていません。
- シーケンスを保証するために、syslog メッセージの本文の部分に追加フィールドが追加されます。このフィールドの内容には、特定のセッションのシーケンス番号が含まれています。複数のセッションで送信される同じメッセージに、それぞれ異なるシーケンス番号が付けられる可能性があります。

syslog セッション

syslog セッションは、デバイス上の syslog エージェントから syslog メッセージの受信者への論理リンクです。たとえば、syslog エージェントと次のいずれかとの間で syslog セッションを確立できます。

- デバイス コンソール
- デバイスのロギング バッファ
- デバイス モニタ

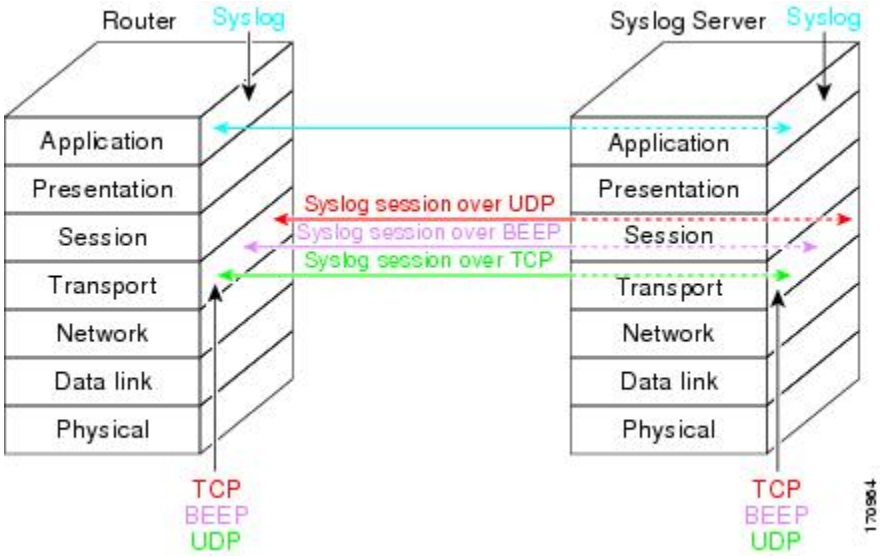
- 外部 syslog サーバ

syslog セッションは、syslog の送信元と syslog の宛先の間の転送接続で動作します。転送接続では次の任意のプロトコルを使用できます。

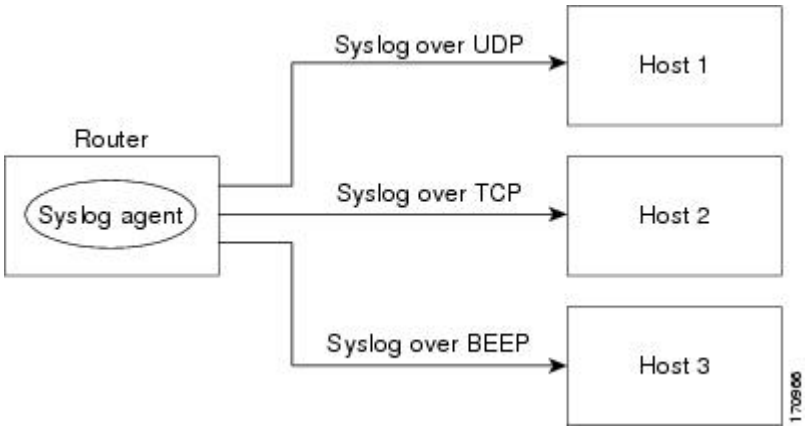
- TCP
- UDP（1 つのリモート アドレスおよびポートとの関連付け）
- BEEP（BEEP セッション内のチャネル）

次の図に、オープン システム相互接続（OSI）モデルを使用したデバイスと syslog サーバ間の syslog セッションおよび転送プロトコルのマッピングを示します。

（注） 次の図は Internet Explorer を使用すると最適に表示されます。



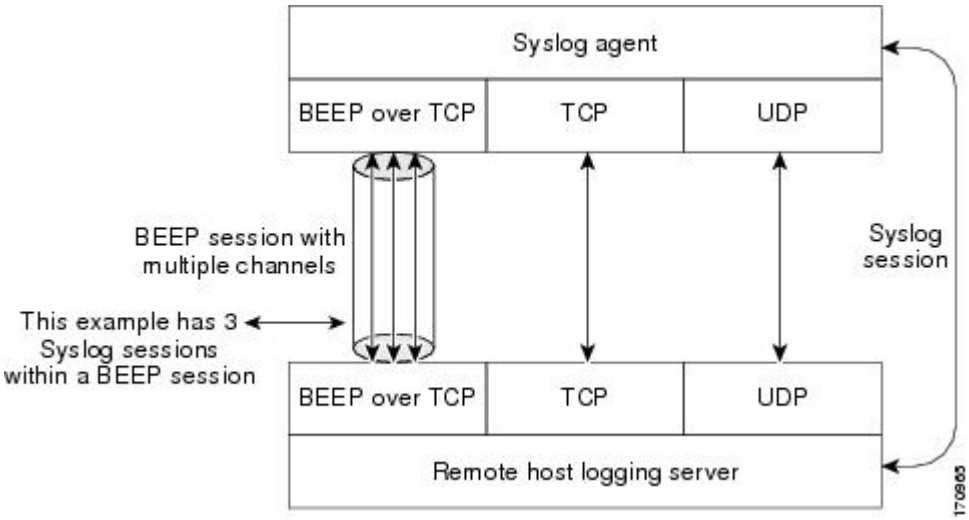
次の図に、UDP、TCP、および BEEP を使用した 1 つの syslog エージェントから複数のホストへの複数の syslog セッションを示します。



複数の syslog セッション

syslog セッションは、転送接続に依存しません。シスコ デバイスは、それぞれが独自の転送接続で動作する複数の syslog セッションをサポートできます。複数の syslog セッションで同じ転送接続を共有できませんが、それぞれ独自の転送接続で動作している複数の syslog セッションを、同じリモート ホストで終端することはできます。1 つの例として、複数のチャネルが使用される BEEP セッションがあります。

次の図は、エンドツーエンドの syslog セッションを示しています。1 つの BEEP セッションに 3 つの syslog セッションがあることに注目してください。



TCP プロトコルと UDP プロトコルには多重チャネルはありませんが、これらのプロトコルでは複数のポートを使用して、syslog ホストへの複数の syslog セッションを確立できます。UDP および TCP 転送方式が BEEP の複数チャネル機能と同様の機能を持つようにするために、syslog の信頼性の高い伝送およびフィルタリング機能では、UDP および TCP 転送方式によって、同じロギング ホストへの複数の syslog セッションを確立できます。BEEP セッションを経由する syslog セッションもサポートされています。

メッセージ ディスクリミネータ

メッセージディスクリミネータはsyslogプロセッサです。メッセージディスクリミネータはsyslogセッションに関連付けられ、セッションを転送接続にバインドします。

メッセージ送信の前に、メッセージは、ユーザが指定した基準リストを持つメッセージ ディスクリミネータの影響を受けます。最初のフィルタリング基準によってメッセージがブロックされると、フィルタリング チェックが停止します。

（注） CLI の基準の順序は、基準がチェックされる順番には影響しません。

- フィルタリング基準は次のとおりです。これらの基準は、次にリストされている順序でチェックされます。
  - 指定された重大度
  - 正規表現と一致するメッセージ本文内のファシリティ
  - 正規表現と一致するニーモニック
  - 正規表現と一致するメッセージ本文の部分

メッセージ ディスクリミネータは、次の機能を提供します。

- オプションのレート制限：超えてはならない、時間間隔あたりのメッセージの転送レートの指定。レート制限を超えた場合、メッセージは、デバイスの判断で遅延するか廃棄されます。レート制限の適用は、その syslog セッションでの syslog メッセージの信頼性の高い伝送が保証されなくなったことを意味します。レート制限の目的は、受信者の syslog サーバで、syslog の保証された伝送を必要としないアプリケーションの「フラッディング」が発生する可能性を回避することにあります。
- 相互関連付け：候補となるイベントメッセージを検査し、イベント全体の情報を可能な限り集約して、集約された情報を含む新しいイベントを作成。関連する機能は次のとおりです。
  - メッセージカウントを維持し、特定のタイプの最初のメッセージの送信とそのタイプの次のメッセージの送信の間の特定の時間待機することによる、重複メッセージの除去。
  - 変動するメッセージの除去
  - 単純なメッセージの相互関連付け。たとえば、あるメッセージが別のメッセージによって報告された原因の症状である場合、1 つの統合されたメッセージが報告されます。

メッセージ ディスクリミネータは、特定の宛先および転送と関連付けることができます。つまり、フィルタはホストに依存する可能性があります。このため、メッセージ ディスクリミネータは、それぞれ異なるディスクリミネータに適用できる複数のセッション、転送、またはチャネルに対して可能なデバイス サポートによって、syslog セッション、転送、またはチャネルに適用されます。

メッセージ ディスクリミネータの確立は、syslog セッションの確立から分離している必要があります。メッセージ ディスクリミネータは、適用される syslog セッション、転送、またはチャネルを参照する必要があります。分離の理由は次のとおりです。

- メッセージ ディスクリミネータは接続から個別に管理でき、メッセージ ディスクリミネータの設定に使用できる機能の調整を syslog セッションの設定方法に反映させる必要はなく、その逆も同様です。
- 複数の接続を同じメッセージディスクリミネータに適用でき、さまざまな syslog 冗長性トポロジが可能です。

明示的なメッセージ ディスクリミネータが syslog セッションと関連付けられていない場合、デバイス全体のグローバル設定から汎用メッセージ ディスクリミネータが使用されます。属性値を指定せずに、「空の」メッセージ ディスクリミネータを作成できます（レート制限もフィルタも設定されません）。

レート制限

Cisco IOS XE syslog でのデバイス全体のレート制限機能は、syslog の信頼性の高い伝送およびフィルタリング機能に保存され、「グローバル レート制限」と呼ばれています。グローバル レート制限を使用しない場合、システム リソースがその量をサポートできる場合には、すべてのイベント メッセージがリモート syslog ホストに送信されます。グローバル レート制限が設定されると、すべての宛先に適用されます。この値は、「汎用メッセージ ディスクリミネータ」のレート制限属性（設定されている場合）に設定されます。グローバル レート制限の欠点は、最も性能の低いリモート syslog ホストのレート制限によって、デバイスが syslog メッセージを送信する速度が設定されることです。

syslog の信頼性の高い伝送およびフィルタリング機能は、syslog セッションベースのレート制限を提供し、グローバル レート制限の影響を回避します。このセッションベースのレート制限は、特定のメッセージ ディスクリミネータと関連付けられており、各 syslog セッションに対して、レート受け入れレベルを別々に設定できます。

セッションベースのレート制限が行われている場合、グローバル レート制限の使用は推奨されません。メッセージディスクリミネータのレート制限は、syslog メッセージの超えてはならないレートを指定しますが、このレートに到達することを保証しません。設定されたグローバル レート制限によって、セッションのレート制限に到達していない場合でも、そのセッションのメッセージが廃棄されることがあります。グローバル レート制限とセッションベースのレート制限が同時に使用される場合、これらの動作について理解することが重要です。

syslog の信頼性の高い伝送およびフィルタリングの利点

- BEEP の認証機能および暗号化機能では、syslog メッセージの信頼性の高いセキュアな伝送が提供されます。
- 基盤となる転送方式に依存しない 1 つのロギング ホストへの複数のセッション
- セッションベースのメッセージ フィルタリングおよびレート制限

- 複数の接続を同じメッセージディスクリミネータに適用でき、さまざまな syslog 冗長性トポロジが可能です。
- デフォルトの syslog カウントを無効にする新しい CLI コマンド
- レート制限によって廃棄される syslog メッセージの関連部分の識別に役立つ新しい CLI コマンド

syslog の信頼性の高い伝送およびフィルタリングの設定方法

メッセージ ディスクリミネータの作成

syslog メッセージのメッセージ ディスクリミネータを作成するには、次の手順を実行します。

手順の概要

1. イネーブル化
2. **configureterminal**
3. **loggingdiscriminator***discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops***string*| **includes***string*}] [**severity** {**drops***sev-num* | **includes***sev-num*}] [**rate-limit***msglimit*]
4. **end**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	イネーブル化 例： `Device> enable`	特権 EXEC モードをイネーブルにします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configureterminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**loggingdiscriminator***discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops***string*| **includes***string*}] [**severity** {**drops***sev-num* | **includes***sev-num*}] [**rate-limit***msglimit*]	ファシリティ サブフィルタを持つメッセージ ディスクリミネータを作成します。 この例では、ファシリティ フィールドに「facl357」があるすべてのメッセージが伝送されます。

	コマンドまたはアクション	目的
	例： `Device(config)# logging discriminator pacfltr1 facility includes facl357`	
ステップ 4	**end** 例： `Device(config)# end`	CLI を特権 EXEC モードに戻します。

メッセージ ディスクリミネータのロギング バッファとの関連付け

メッセージ ディスクリミネータを特定のバッファと関連付けるには、次の作業を実行します。

手順の概要

1. イネーブル化
2. **configureterminal**
3. **loggingdiscriminator***discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops***string*| **includes***string*}] [**severity** {**drops***sev-num* | **includes***sev-num*}] [**rate-limit***msglimit*]
4. **loggingbuffered** [**discriminator***discr-name* | **xml**] [*buffer-size*] [*severity-level*]
5. **end**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	イネーブル化 例： `Device> enable`	特権 EXEC モードをイネーブルにします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configureterminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**loggingdiscriminator***discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops***string*| **includes***string*}]	メッセージディスクリミネータを作成します。

	コマンドまたはアクション	目的
	[**severity** {**drops***sev-num* | **includes***sev-num*}] [**rate-limit***msglimit*] 例： `Device(config)# logging discriminator pacfltr2`	
ステップ 4	**loggingbuffered** [**discriminator***discr-name* | **xml**] [*buffer-size*] [*severity-level*] 例： `Device(config)# logging buffered discriminator pacfltr2 5`	ローカル バッファへのロギングをイネーブルにし、メッセージ ディスクリミネータを指定します。
ステップ 5	**end** 例： `Device(config)# end`	CLI を特権 EXEC モードに戻します。

メッセージ ディスクリミネータのコンソール端末との関連付け

メッセージ ディスクリミネータをコンソール端末と関連付けるには、次の作業を実行します。

手順の概要

1. イネーブル化
2. **configureterminal**
3. **loggingdiscriminator***discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops***string*| **includes***string*}] [**severity** {**drops***sev-num* | **includes***sev-num*}] [**rate-limit***msglimit*]
4. **loggingconsole** [**discriminator***discr-name* | **xml**] [*severity-level*]
5. **end**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	イネーブル化 例： `Device> enable`	特権 EXEC モードをイネーブルにします。 • パスワードを入力します（要求された場合）。

	コマンドまたはアクション	目的
ステップ 2	**configureterminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**loggingdiscriminator***discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops***string*| **includes***string*}] [**severity** {**drops***sev-num* | **includes***sev-num*}] [**rate-limit***msglimit*] 例： `Device(config)# logging discriminator pacfltr3`	メッセージ ディスクリミネータを作成します。
ステップ 4	**loggingconsole** [**discriminator***discr-name* | **xml**] [*severity-level*] 例： `Device(config)# logging console discriminator pacfltr3 1`	コンソールへのロギングをイネーブルにして、特定の重大度のメッセージをフィルタリングするメッセージ ディスクリミネータを指定します。
ステップ 5	**end** 例： `Device(config)# end`	CLI を特権 EXEC モードに戻します。

メッセージ ディスクリミネータの端末回線との関連付け

メッセージディスクリミネータを端末回線に関連付け、モニタにメッセージを表示するには、次の手順を実行します。

手順の概要

1. イネーブル化
2. **configureterminal**
3. **loggingdiscriminator***discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops***string*\| **includes***string*}] [**severity** {**drops***sev-num* \| **includes***sev-num*}] [**rate-limit***msglimit*]
4. **loggingmonitor** [**discriminator***discr-name*\| **xml**] [*severity-level*]
5. **end**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	イネーブル化 例： `Device> enable`	特権 EXEC モードをイネーブルにします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configureterminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**loggingdiscriminator***discr-name* [[**facility**] [**mnemonics**] [**msg-body**] {**drops***string*| **includes***string*}] [**severity** {**drops***sev-num* | **includes***sev-num*}] [**rate-limit***msglimit*] 例： `Device(config)# logging discriminator pacfltr4`	メッセージ ディスクリミネータを作成します。
ステップ 4	**loggingmonitor** [**discriminator***discr-name*| **xml**] [*severity-level*] 例： `Device(config)# logging monitor discriminator pacfltr4 2`	pacfltr4 という名前のメッセージ ディスクリミネータを指定し、重大度 2 以下で端末回線へのメッセージのロギングをイネーブルにします。
ステップ 5	**end** 例： `Device(config)# end`	CLI を特権 EXEC モードに戻します。

メッセージ カウンタのイネーブル化

デバッグ、ログ、または syslog メッセージのロギングをイネーブルにするには、次の手順を実行します。

手順の概要

1. イネーブル化
2. **configureterminal**
3. **loggingmessage-counter**{**debug** | **log** | **syslog**}
4. **end**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	イネーブル化 例： `Device> enable`	特権 EXEC モードをイネーブルにします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configureterminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**loggingmessage-counter**{**debug** | **log** | **syslog**} 例： `Device(config)# logging message-counter syslog`	syslog メッセージのロギングをイネーブルにします。
ステップ 4	**end** 例： `Device(config)# end`	CLI を特権 EXEC モードに戻します。

BEEP セッションの追加と削除

BEEP セッションを追加および削除するには、次の手順を実行します。

手順の概要

1. イネーブル化
2. **configureterminal**
3. **logginghost** {{*ip-address* | *hostname*} [**vrf***vrf-name*] | **ipv6**{*ipv6-address* | *hostname*}} [**discriminator***discr-name* | [[**filtered** [**stream***stream-id*] | **xml**]] [**transport** {[**beep** [**audit**] [**channel***chnl-number*] [**sasl***profile-name*] [**tlscipher** [*cipher-num*] **trustpoint***trustpt-name*]]] | **tcp**[**audit**] | **udp**} [**port***port-num*]] [**sequence-num-session**] [**session-id**{**hostname** | **ipv4** | **ipv6** | **string***custom-string*}]
4. **end**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	イネーブル化 例： `Device> enable`	特権 EXEC モードをイネーブルにします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configureterminal** 例： `Device# configure terminal`	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**logginghost** {{*ip-address* | *hostname*} [**vrf***vrf-name*] | **ipv6**{*ipv6-address* | *hostname*}} [**discriminator***discr-name* | [[**filtered** [**stream***stream-id*] | **xml**]] [**transport** {[**beep** [**audit**] [**channel***chnl-number*] [**sasl***profile-name*] [**tlscipher** [*cipher-num*] **trustpoint***trustpt-name*]]] | **tcp**[**audit**] | **udp**} [**port***port-num*]] [**sequence-num-session**] [**session-id**{**hostname** | **ipv4** | **ipv6** | **string***custom-string*}] 例： `Device(config)# logging host host3 transport beep port 600 channel 3`	ロギング ホストを指定し、メッセージのロギングのための転送プロトコル、ポート、およびチャネルを指定します。
ステップ 4	**end** 例： `Device(config)# end`	CLI を特権 EXEC モードに戻します。

syslog の信頼性の高い伝送およびフィルタリングの設定例

転送とロギングの設定例

```
Device(config)# do show running-config
 | include logging
logging buffered xml
logging host 209.165.201.1 transport udp port 601
Device(config)# logging host 209.165.201.1 transport beep port 600 channel 3
Device(config)# logging host 209.165.201.1 transport tcp port 602

Device(config)# show running-config | include logging
logging buffered xml
logging host 209.165.201.1 transport udp port 601
logging host 209.165.201.1 transport beep port 600 channel 3
logging host 209.165.201.1 transport tcp port 602
Device(config)#
```

syslog トランザクションの VRF 対応送信元インターフェイスに関する追加情報

関連資料

関連項目	マニュアル タイトル
Cisco IOS コマンド	『*Cisco IOS Master Commands List, All Releases*』
ネットワーク管理コマンド（logging コマンドを含む）：コマンド構文の詳細、デフォルト設定、コマンドモード、コマンド履歴、使用上のガイドライン、および例	『*Cisco IOS Network Management Command Reference*』
Syslog ロギング	「*Troubleshooting and Fault Management*」モジュール

標準規格および RFC

規格/RFC	タイトル
この機能によりサポートされる新規または変更された標準/RFC はありません。またこの機能による既存の標準/RFC のサポートに変更はありません。	--

MIB

MIB	MIB のリンク
この機能によってサポートされる新しい MIB または変更された MIB はありません。またこの機能による既存 MIB のサポートに変更はありません。	選択したプラットフォーム、Cisco ソフトウェア リリース、およびフィーチャ セットの MIB を検索してダウンロードする場合は、次の URL にある Cisco MIB Locator を使用します。 http://www.cisco.com/go/mibs

シスコのテクニカル サポート

説明	リンク
★枠で囲まれた Technical Assistance の場合★右の URL にアクセスして、シスコのテクニカル サポートを最大限に活用してください。これらのリソースは、ソフトウェアをインストールして設定したり、シスコの製品やテクノロジーに関する技術的問題を解決したりするために使用してください。この Web サイト上のツールにアクセスする際は、Cisco.com のログイン ID およびパスワードが必要です。	http://www.cisco.com/cisco/web/support/index.html

syslog の信頼性の高い伝送およびフィルタリングの機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコ ソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

表 *3*：*syslog* の信頼性の高い伝送およびフィルタリングの機能情報

機能名	リリース	機能情報
syslog の信頼性の高い伝送およびフィルタリング	Cisco IOS XE Release 2.1	syslog の信頼性の高い伝送およびフィルタリング機能によって、デバイスを syslog メッセージの受信用にカスタマイズできます。この機能は、BEEP を使用した syslog メッセージの信頼性の高いセキュアな伝送を提供します。さらに、基盤となる転送方式にかかわらず、1 つのロギング ホストへの複数のセッションを可能にし、メッセージ ディスクリミネータと呼ばれるフィルタリング メカニズムを提供します。 Cisco IOS XE リリース 2.1 では、この機能は Cisco ASR 1000 シリーズ アグリゲーション サービス ルータに導入されました。 導入または変更されたコマンドは、[illegible] です。